Trust Management System for Opportunistic Cloud Services

Kuada, Eric

Published in:
2nd International Conference on Cloud Networking (CloudNet). San Francisco, USA: IEEE.

DOI (link to publication from Publisher):
10.1109/CloudNet.2013.6710555

Publication date:
2013

Document Version
Early version, also known as pre-print

Trust Management System for Opportunistic Cloud Services

Eric Kuada
Department of Electronic Systems
Aalborg University, Copenhagen, Denmark
kuada@cmi.aau.dk

Abstract - We have over the past three years been working on the feasibility of Opportunistic Cloud Services (OCS) for enterprises. OCS is about enterprises strategically contributing and utilizing spare IT resources as cloud services. One of the major challenges that such a platform faces is data security and trust management issues. This paper presents a trust management system for OCS platforms. It models the concept of trust and applies it to OCS platforms. The trust model and the trust management system are verified through the simulation of the computation of the trust values with Infrastructure as a Service, and Software as a Service, usage scenarios.

Keywords: Opportunistic Cloud Services; Trust Management System; Trust Engineering; Pseudo Service Level Agreement

I. INTRODUCTION

This introductory section begins with the motivation for this study, and then is followed by the background for trust engineering efforts in cloud computing.

A. Motivation for the Study

We have over the past three years been working on the feasibility of Opportunistic Cloud Services (OCS) for enterprises [1] [2] [3]. We have been working on the feasibility of its successful implementation in terms of the technical feasibility, impact of public policy and regulations on its implementation [3] , and the development of suitable incentive mechanisms for OCS networks [2]. We have also been working on the role of cloud computing for Information Technology (IT) resource cost management for SMEs [4], and how to leverage opportunistic cloud services to lighten the IT resource needs of SMEs, particularly in developing economies.

One of the major challenges that such a platform faces is data security and trust management issues. The OCS network platform is a governing platform that serves as the social networking platform for enterprises and also includes interoperable cloud management tools with which member enterprises can provide resources that will be used by other enterprises interested in these services. Members normally will package only their spare IT resources and make them available as Cloud services on the OCS platform so that others interested can utilize them. For example a member institution that has virtualized its data center into a private cloud can plan its resources such that it can make some of these resources available to the OCS platform.

Since no business agreement and hence no Service Level Agreement (SLA) exist between the service providers and the potential users of their services, service consumers do not enjoy the level of support (in terms of quality of service, reliability, availability, security, billing transparency, etc.) that commercial cloud service providers offer to their clients. Considering the fact that commercial cloud service providers are finding it extremely challenging to provide such a support, together with providing adequate transparency in their services to their clients with the hope of establishing the necessary trust, the OCS platform more so needs a well-crafted and soundly engineered trust management system in order to make resources on the platform suitable for business use.

B. Trust Engineering in Cloud Computing

Cloud computing is essentially the packaging of traditional information technology infrastructure and software solutions such as storage, CPU, network, applications, services, etc. as virtualized resources and delivered by a service provider to its customers as an on-demand pay-per-use self-provisioned service; normally offered through a web portal over a network such as the Internet [5] [6] [7]. As vendors start to deploy Cloud services, and users upload data in the Cloud to utilize them, a new privacy concern arises, because data owners would like to preserve the confidentiality of their data, and under some circumstance even their identities private from the software provider. While cloud service providers pledge to preserve data privacy, the current Software as a Service (SaaS) architecture makes it difficult to provide any assurance that the software in the Cloud will not be able to make copies or redistribute the data it used [8] . The Cloud model is based on two key characteristics: multi-tenancy, where multiple tenants share the same service instance, and elasticity, where tenants can scale the amount of their allocated resources based on current demands. Although both characteristics target improving resource utilization, cost and service availability, these gains are threatened by multi-tenancy security implications. The sharing of applications that process critical information without sufficient proven security isolation, security SLAs or tenant control, results in "loss-of-control" and "lack-of-trust" problems [9].

Apart from these consumer concerns, cloud architectures also introduce new classes of security risks and attacks over the resources of cloud service providers. These include poisoned virtual machines, attacks against the Cloud Service Provider (CSP) management console, attacks based on knowledge of default security settings, abuse of billing systems, attacks that abuse the trust associated with the CSP's namespace, and data leakage via uniform resource locators. Currently, CSPs do not have robust technical solutions that can protect their cloud resources from harmful malware, virus infection, botnets, distributed denial of service attacks, or other types of cyber-attacks. Furthermore, there is no effective mechanism to help cloud users evaluate the security measures of their service providers and ensure the protection of their data while taking into consideration industry standards or personal preferences [9].

In order to design and develop a trust management system for the OCS platform, we needed to look at the current trust engineering issues in cloud computing. It was decided that we needed to perform a systematic literature review on this topic because since the OCS concept itself is new, any trust design models of its subsystems must be based on and guided by exhaustive knowledge of the state-of-the-art in the field. Some of the main findings of the study are that, employing trusted computing technologies and reputation based approaches are two key approaches to trust engineering in the cloud computing marketplace. Also, trusted third party approaches and the deployment model play a significant part in enhancing trust between service providers and their consumers. Based on the findings during the study, the main areas of trust engineering research focus has been on quality of service, security, access and identity management, user support on trust management,

and accountability in in the context of a cloud computing marketplace . Though the objective of [10] is a formal trust specification which covers a wide range of intuitive trust characteristics such as trust transitivity and mutual relationship, much work has not been in this area of formal trust modeling. According to[11], discussions about cloud computing security often fail to distinguish general issues from cloud-specific issues. A similar trend is seen in the discussion of trust in cloud computing and trust engineering in general where the concept of trust in treated loosely without any formal specification or definition in its treatment. Formal trust modeling and definitions are however very necessary in ensuring a unified view of the concept of trust in the design and engineering of trust management systems for cloud computing. As a first step towards addressing this problem, we contextualized the formal trust specification of multi-agent environments for cloud computing environments, and provided a formal definition of the concept of trust as is applicable to the cloud computing marketplace.

The rest of the paper is organized as follows: Section II models the concept of trust for OCS platforms. Section III presents the design of a trust management system for OCS platforms. This is followed with the OCS trust management architecture in Section IV. Section V presents work on the verification of the trust model and the designed trust management system. Section VI concludes the paper and also touches on some insights on future work.

II. TRUST MODEL FOR OCS PLATFORMS

Opportunistic Cloud Services (OCS) is a social network approach to the provisioning and management of cloud computing services for enterprises; it deals with the concept of enterprises taking advantage of cloud computing services to meet their business needs without having to pay or paying a minimal fee for the services. The OCS network is modelled and implemented as a social network of enterprises collaborating strategically for the contribution and usage of cloud computing services without entering into any business agreements.

1) Nature of Members and Services

An OCS network consists of a set of strategic members contributing and utilizing cloud computing services. The platform consists of a set of services each belonging to a category; each service has a non-monetary cost that varies dynamically. The service or resource contributed by a member is of a certain finite capacity and the resources to a particular service may be contributed by multiple members. Members will normally only contribute resources that they have spare capacity of (e.g. CPU, storage, application that they have developed internally, etc.). That is, they package their spare IT resources as cloud services and make them available to the OCS platform. Members are free to provide and discontinue one or more services at will at any point in time. They are likewise free to use or discontinue the usage of one or more services at will at any point in time [2].

2) Trust Model in the context of OCS

Though there has been some work on trust modeling and trust management systems, and even in the new domain of trust management systems for cloud computing environments [12] [13] [14], the subjective nature of trust has made a solid definition elusive. Researchers have most often used the term loosely in their work; more specifically, a rigorous formal definition has not been applied in most cases.

The adopted definition and model of the concept of trust in this work is an adaptation of [15]: The level of trust, $T_c^p(t_i)$ of a service consumer c for a service provider p in the context of a transaction $t_i \in T$ is the a priori probability that the utility of c will meet or exceed its minimum threshold of satisfaction u_0 at the end of transaction t_i, given c 's perceived trustworthiness of service provider p. Simply stated, trust is the level of confidence of c that the outcome of a transaction with another agent p will be satisfactory for it. More formally:

$T_c^p(t_i) = \int_{U_c(R) \geq u_0} \tau_c^p(R, t_i).dR$, where $U_c(R)$ is the utility function of service consumer c; and $\tau_c^p(R, t_i)$ - the trustworthiness of service provider p as perceived by consumer c in the context of a transaction $t_i \in T$ is the a priori subjective joint probability distribution function of the critical rating vector $R_c^p(t_i)$ from the perspective of c. This definition and model of trust has been adopted by this paper because it provides this formal specification which is necessary in ensuring a unified view of the concept of trust in the design and engineering of trust management systems for cloud computing.

It is not only cloud service consumers that need the consideration of trust in their transactions with the cloud service providers. Most often than not, cloud services providers also need to be wary of the activities of cloud service consumers. Thus, trust modeling is useful in the analysis of the genuine and potentially malicious service consumers. Therefore a trust model is needful for the perceived trustworthiness of service consumers by the providers of the services. So similarly, the level of trust $T_p^c(t_i)$ of a service provider p for a service consumer c in the context of a transaction $t_i \in T$ is the a priori probability that the utility of p will meet or exceed its minimum threshold of satisfaction u_0 at the end of transaction t_i, given service provider p 's perceived trustworthiness of service consumer c. Again, more formally: $T_p^c(t_i) = \int_{U_p(R) \geq u_0} \tau_p^c(R, t_i).dR$, where $U_p(R)$ is the utility function of service provider p; and $\tau_p^c(R, t_i)$ - the trustworthiness of service consumer c as perceived by service provider p in the context of a transaction $t_i \in T$ is the a priori subjective joint probability distribution function of the critical rating vector $R_p^c(t_i)$ from the perspective of p. Please note that it is for notational simplicity that the critical rating vectors $R_c^p(t_i)$ and $R_p^c(t_i)$ are denoted by R (without the full complement of the subscripts) in the denotation of the trustworthiness.

The above definitions have a number of interesting properties which correspond with the intuitive properties of trust in our everyday life such as trustworthiness is subjective, and it is defined relative to a particular set of critical attributes; trustworthiness is defined at a given point in time, and it is defined as a probability distribution. Some other important intuitive attributes of trust are that trust has duality - it is subjective and objective; that is some of the critical attributes are subjectively measureable and others are objectively measureable; trust is not always symmetrical, that is, A trusts

B, does not always mean B trusts A; and trust is dynamic, that is, trust is related to context and temporal factors [16].

We now consider the level of trust in a service - this time not the service providers but rather the services themselves. Since a particular service may come into fruition as a combination of resources and services from multiple providers, each service's trust level must be assessed as an autonomous entity even though this trust level is a function of the composite trust level of the providers and the base services from which it has been derived. So we define the trust level of a service s as:

The level of trust $T_c^s(t_i)$ of a service consumer c in a service s in the context of a transaction $t_i \in T$ is the a priori probability that the utility of c will meet or exceed its minimum threshold of satisfaction u_0 at the end of transaction t_i, given $c's$ perceived trustworthiness of service s.

$T_c^s(t_i) = \int_{U_c(R) \geq u_0} \tau_c^s(R, t_i).dR$, where $U_c(R)$ is the utility function of service consumer c; and $\tau_c^s(R, t_i)$ - the trustworthiness of service s as perceived by consumer c in the context of a transaction $t_i \in T$ is the a priori subjective joint probability distribution function of the critical rating vector $R_c^s(t_i)$ from the perspective of c.

The application of this trust modeling for the OCS platform and its usage will be demonstrated in Sections IV and V which respectively discuss the components of the OCS trust management architecture, and the OCS trust model verification by demonstrating how it is applied to an IaaS and SaaS usage scenarios.

C. Trust Production Approaches

There are generally three basic ways by which communities of interacting entities go about the issue of trust generation. These are: norms backed up by institutional guarantees, indirect cues, and reputational information [15]. *Norms and institutional guarantees* attempt to reduce the uncertainty on the behavior of other agents by prescribing specific allowed behavioral ranges (which usually correspond to satisfactory outcomes for the majority of transaction types and society members) and by providing institutions, which prevent deviations or make such deviations highly unlikely because of quick detection and effective sanctions [17]. *Indirect cues* are attributes of an agent, which we have associated with certain likely behaviors based on our experience, intuition and training. *Reputational information* is information about, or observations of an agent's past behavior on similar situations that is aggregated and distributed by means of word-of-mouth or through trusted third parties, such as credit rating agencies, consumer reports, etc. Reputational information can help agents construct estimates on another agent's trustworthiness under the assumption that agents have an underlying distribution of behavior, which is relatively stable over time[15].

Trust production through indirect cues is however inapplicable in online communities because this is mostly obtained by interacting agents through observing the body language and appearance of the other party they are transacting with. The application of institutional guarantees is also normally infeasible in cyberspace because of the anonymity enjoyed by the interacting entities and the lack of appropriate institutions or their inability to enforce adequate guarantees. However, findings from our previous work has shown that the major approaches to trust engineering in cloud computing are that, employing trusted computing technologies and reputation based approaches are two key approaches to trust engineering in the cloud computing marketplace. Also trusted third party approaches and the deployment model play a significant part in enhancing trust between service providers and their consumers. In the context of trust engineering for OCS environments, two of these come handy. These are the trusted third party based and the reputation based approaches.

1) Trusted Third Party Based

The OCS platform by its very nature can serve as a trusted third party for enhancing trust between service contributors and the utilizers of these services by providing and enforcing OCS institutional guarantees for the allowed normal behavior on the platform. This could be achieved through for example standardized Pseudo SLA templates (see **Section IV** for the discussion on the implementation and management of the Pseudo SLA system and its associated templates), and the detection of problematic services, service providers, and service consumers. Appropriate actions can then be taken to subsequently penalize the offending entities.

2) Reputation Based

The platform also lends itself to the gathering, management, and analysis of reputation information about services, service providers and service consumers. This information can then be used in supporting entities on the platform in making decisions about the trustworthiness of other entities they are interested in transacting with.

3) Trust Values

The platform will provide the following trust values to its members in facilitating their decision making process. The trust values of interest are: trust value of a service or a resource, trust value of the service category to which the service/resource belongs, trust value of service/resource providers, composite trust value of the group to which service /resource provider belongs, and trust value of resource consumers. This list is however extensible as new needs concerning trust values arise.

III. OCS TRUST MANAGEMENT SYSTEM

Some of the desired properties of the OCS trust management system are reliability, robustness, scalability, and usability. The verification of the system in this work will however focus on its applicability and usability. Demonstration of the other desired properties will be part of our future work when the responsible components are fully implemented.

A. Main Elements of trust management systems

The generic operations of trust management include expectation, data monitoring, data management, analysis, and decision making.

1) Expectation:

This is a function of the service user's requirement specification, and service provider's declared intensions of supported quality of service. This is normally expressed in SLAs that the cloud service provider and their customers enter into. There is a need for expectation management from both ends to ensure smooth arbitration in case of any breaches in the agreement. Since no formal agreement exists between the service providers and the users of these services, this expectation management is expressed on the OCS platform as Pseudo SLA.

We adapt some of the stages of the work of [18] on the usage of public SLA templates in cloud markets to develop the creation and usage of pseudo SLA templates for the OCS platform. The implementation and management of the pseudo SLA concept on the OCS platform is in three main parts: the creation of pseudo SLA templates for categories of services by the OCS platform, the pseudo SLA specification by service providers, and the quality requirement specification by potential service users. These main parts are respectively outlined below.

a) Existing OCS platform pseudo SLA templates

i. Platform administrators create pseudo SLA (pSLA) templates for service categories and their sub-categories. This normally will be based on the cloud service type such as IaaS, PaaS, SaaS, DaaS, etc.
ii. Request SLAs (rSLAs) may be promoted to OCS pSLAs by the OCS platform administrators
iii. Each pSLA is stored in the SLA repository.

b) Service providers pseudo SLA specification

The service providers' declared intentions are specified in service SLAs.

i. The service provider specifies the category of service (and may also specify some critical attribute values), and it is then presented with a list of existing pSLA templates that the OCS platform considers as suitable templates.
ii. The service provider assigns the service to a pSLA template.
iii. The service provider creates the service SLA (sSLA) for the service by accepting a presented pSLA template or modifying it to create the sSLA to meet the specification of this service.
iv. The new service SLA (sSLA) is stored in the service SLA repository.

c) Service users' quality requirements specification

i. A potential service user specifies the service category together with the values of some critical attributes of the potential service to use.
ii. The user is presented with possible matching sSLAs from the service SLA repository.
iii. The user may accept one of these sSLAs and proceed to use the service or create a request SLA (rSLA) based on one of the sSLAs to suit its usage requirements.
iv. The new rSLA is stored in the request SLA repository which can be presented as an OCS pSLA template to service providers during the service SLA creation stage of the service creating process.

Fig.1 shows the processes involved in the pseudo SLA implementation and management.

2) Data Monitoring and Management

The data monitoring and management is concerned with what kinds of data should be monitored and the associated cost of monitoring. It is therefore responsible for defining new trust data that need to be monitored on the OCS platform. The kind of data that should be monitored should facilitate the gathering of information for computing the necessary trust values. These are the trust value of services and the trust value of the service category to which they belong; trust value of service providers, trust value of service consumers, and the composite trust value of the group to which service providers belong. Examples of some of the data that needs to be monitored in the case of computing the trust values of services are: how long the service has been in operation, percentage service uptime, probability distribution of service failures, and user ratings of the service. Some of the important parameters that need monitoring in the case of service providers are: how long the provider has been providing services, the services that the provider has been providing, trust values of all the services the provider has been providing, and the probability distribution of the failure of its services. Some of the parameters of monitoring interest in computing trust value of service consumers are service usage patterns, and reports of malicious behavior.

The data management also deals with defining data storage policies such as for example local storage of trust matrix by members, storage of member interactions by the OCS platform, the type of communication and exchange of information, and what type of data are to be exchanged.

3) Data Analysis

The data analysis is concerned with the computation of the requisite trust values based on the information from the expectation of consumers and providers of services.

a) Cloud Computing Parameters of Trust

When selecting a cloud service provider, multiple important parameters that are of relevance to the cloud service consumer need to be identified properly. Also, there is need for mechanisms to measure those parameters and aggregate these measurements based on the customers' preference regarding the importance of the parameters[19]. References [20] and [19] have identified several of these parameters which have been categorized into quality of service related, security and privacy related, risk management related, and reputation related attributes. These parameters (attributes) are termed critical attributes; more formerly, a *critical attribute* of a service provider s, from the perspective of a service consumer c, in the context of a transaction $t_i \in T$, is an attribute whose value affects the utility of c and is contingent upon the behavior of s in the course of transaction t_i [15].

b) Reputation System (computation of reputation)

The trustworthiness, $\tau_c^p(R, t_i)$ (reputation, if trustworthiness is from only reputational information) of service provider p as perceived by consumer c in the context of a transaction $t_i \in T$ is the a priori subjective joint probability distribution function of the critical rating vector $R_c^p(t_i)$ from the perspective of c. Instead of a single value rating, we have rating of metrics of intent, integrity, capability and results of the critical attributes of the entity to be trusted, from which our model computes the trustworthiness; trust values are then computed from this. That is, for each critical attribute (e.g. accuracy, reliability, etc.) in the critical rating vector $R_c^p(t_i)$ that has been identified from the rSLA of the service consumer, the reputational ratings are based on the intent, integrity, capability and results; where intent constitutes information about declared agendas about what entities promise to provide through their services. Integrity constitutes information about honesty; this is a measure of, to what extent entities deliver on what they promised. Capability constitutes information about owned or outsourced resources (what assets parties have); and finally, results constitute information about products and services that entities specialized in through consistently delivering these products and services satisfactorily to their clients [21].

4) Decision Support

The purpose of the users of a trust management system is to make decisions concerning engaging in particular transaction at a point in time. It is imperative to provide an intuitive representation of trust values for all the types of users of the OCS platform. This will be handled by the Decision Support Manager as is discussed later in the next section.

IV. OCS TRUST MANAGEMENT ARCHITECTURE

The major components of the OCS trust management system are Expectation Manager (EM), Platform Guarantees Enforcement Manager (PGEM), Data Monitoring Manager (DMoM), Data Management Manager (DMaM), Trust Analysis Manage (TAM), and Decision Support Manager (DSM). Fig. 2 shows the relationship between these components.

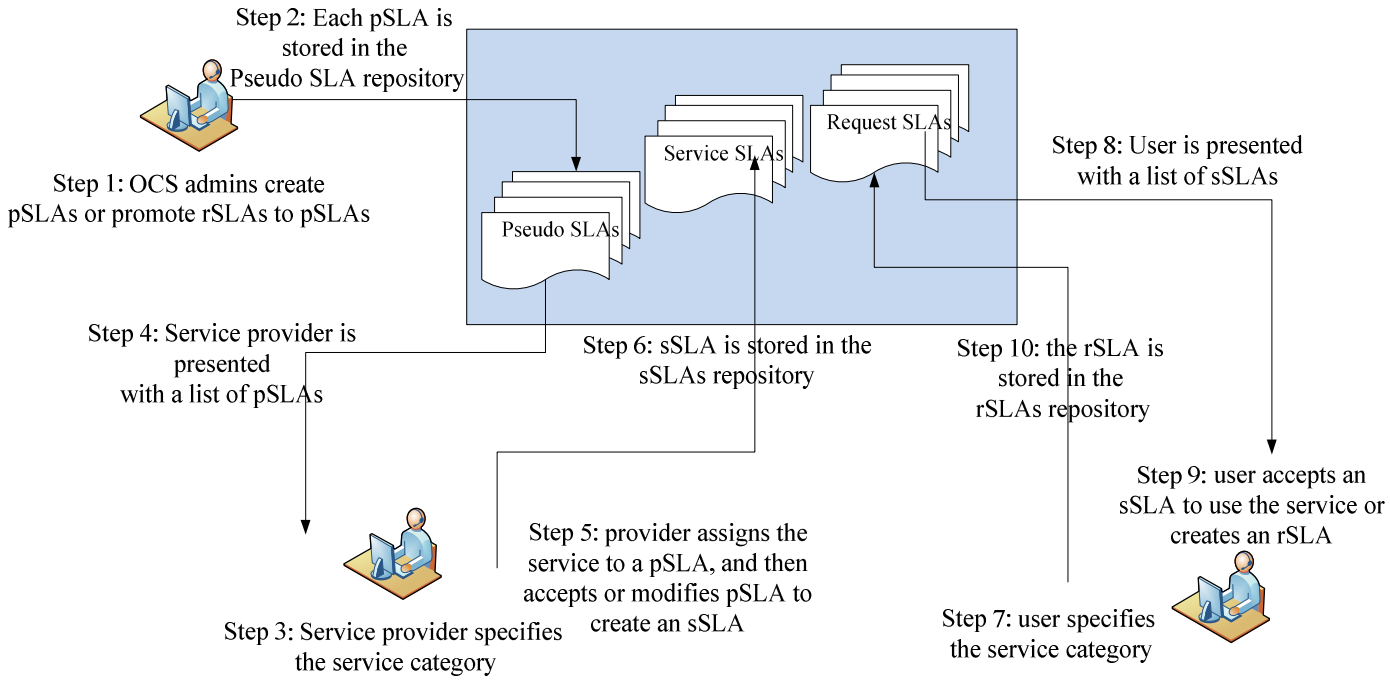

Fig. 1: Processes involved in the pseudo SLA implementation and management

A. Expectation Manager

The expectation manager is responsible for handling the creation and maintenance of the OCS Platform pseudo SLA (pSLA) templates, the service provider assignment of services to a particular pSLA template, and the creation of service SLA (sSLA) to meet the specification of each service. It is also responsible for presenting potential service users with possible matching sSLAs from the service SLA repository which they may accept and proceed to use the service or create a request SLA (rSLA) based on one of the sSLAs to suit their specific usage requirements.

B. Platform Guarantees Enforcement Manager

This module is responsible for ensuring good and acceptable behavior on the platform. It applies appropriate sanctions to undesirable behaviors on the platform. It is therefore responsible for malicious conditions detection and the detection of SLA violation, and then taking appropriate remedial actions such as removing offending services from the platform and banning offending users.

C. Data Monitoring Manager

The DMoM is responsible for defining new trust data that needs to be monitored on the OCS platform in order to accommodate for adapting the platform to future needs such as when new service categories and categories of trust values are needed to be computed.

D. Data Management Manager

The data management manager is responsible with defining data storage policies such as for example local storage of trust matrix by members, storage of member interactions by the OCS platform, the type of communication and exchange of information, and what type of data are to be exchanged. It also deals with data reliability, security, recovery in case of problems and maintaining consistency in situations of discrepancies in data from multiple sources.

E. Trust Analysis Manager

This module makes the analyses of the trust values to be computed from information from the expectation manager and the available data from the data management module. It then computes the necessary trust values. It is evident from our model of the definition of trust on the OCS platform that we are interested in personalized trust values for the entities on the OCS platform. For example a potential service consumer will be interested in computing its perceived trust value for a service and its perceived trust value of the provider(s) of that service. This is achieved with equations (1) and (2), and the trust values computation algorithm as described below.

The level of trust $T_c^p(t_i)$ of a service consumer c for a service provider p in the context of a transaction $t_i \in T$ is the a priori probability that the utility of c will meet or exceed its minimum threshold of satisfaction u_0 at the end of transaction t_i, given $c's$ perceived trustworthiness of service provider p.

$T_c^p(t_i) = \int_{U_c(R) \geq u_0} \tau_c^p(R, t_i).dR$, ------- (1) where $U_c(R)$ is the utility function of service consumer c; and $\tau_c^p(R, t_i)$ - the trustworthiness of service provider p as perceived by consumer c in the context of a transaction $t_i \in T$ is the a priori subjective joint probability distribution function of the critical rating vector $R_c^p(t_i)$ from the perspective of c.

The level of trust $T_c^s(t_i)$ of a service consumer c for a service s in the context of a transaction $t_i \in T$ is the a priori probability that the utility of c will meet or exceed its minimum threshold of satisfaction u_0 at the end of transaction t_i, given $c's$ perceived trustworthiness of service s.

$T_c^s(t_i) = \int_{U_c(R) \geq u_0} \tau_c^s(R, t_i).dR$, ------- (2) where $U_c(R)$ is the utility function of service consumer c; and $\tau_c^s(R, t_i)$ - the trustworthiness of service s as perceived by consumer c in the context of a transaction $t_i \in T$ is the a priori subjective joint probability distribution function of the critical rating vector $R_c^s(t_i)$ from the perspective of c.

1) Trust Values Computation Algorithm

1. Identify service dependencies from rSLA
2. Compute trust value of the service based on eq. (2)
3. Compute trust value (based on eq. (1)) of each of the service providers contributing to this service
4. Compute composite trust value, $T_c^{comp}(t_i)$ based on

$$T_c^{comp}(t_i) = \omega_s T_c^s(t_i) + \omega_{p1} T_c^{p1}(t_i) + \omega_{p2} T_c^{p2}(t_i) + \ldots + \omega_{pn} T_c^{pn}(t_i)$$

$$where, \omega_s + \omega_{p1} + \omega_{p2} \ldots + \omega_{pn} = 1$$

5. For each service dependencies in step 1, repeat steps 2, 3 and 4
6. Compute the overall composite trust value by applying the appropriate dependency level weight(ω_l) based on the level of the dependency in the dependency chain for all composite trust values as in step 4

$$T_c^{total}(t_i) = \omega_1 T_1^{comp}(t_i) + \omega_2 T_2^{comp}(t_i) + \ldots + \omega_l T_l^{comp}(t_i),$$

$$\text{where } \omega_1 + \omega_2 \ldots + \omega_l = 1$$

To compute $T_c^{comp}(t_i)$ from equations (1) and (2), we need the utility function $U_c(R)$, the critical rating vectors $R_c^p(t_i)$ and $R_c^s(t_i)$, the weights(ω_s, ω_p and ω_l), and the trustworthiness (reputation) distribution functions $\tau_c^p(R, t_i)$ and $\tau_c^s(R, t_i)$. The critical rating vectors are internal properties of c which is specified in or can be derived from its rSLA; and the utility function is also an internal property of c which it can provide for the algorithm to compute the trust values. All the weights, ω_s, ω_p and ω_l, are parameters that are determined by the OCS platform. The remaining required parameters are the reputation distribution functions; and these are also available from the data monitoring and data management modules on the OCS platform as reputation data. The only issue left is the need to convert the reputation data that have been collected into standardized probability distribution functions to simplify the trust value computations and provided tractable solutions.

2) Approximation of reputation (trustworthiness)

The approximation of the reputation data to standard known probability distribution functions (e.g. Uniform distribution, normal distribution, etc.), can be achieved using appropriate curve fitting algorithms. The reputation data gathered by the DMoM and DMaM modules can be approximated to standard probability distributions function using curve fitting algorithms such as polynomial (linear, quadratic, 3rd order, etc.), and logarithmic algorithms.

F. Decision Support Manager

The decision support manager is responsible for taking results from the trust value computations of the analysis manager and presenting it in a format that simplify visualization for the users. The user-friendly trust value representation together with making recommendations on decisions to be taken by users should facility their decision making process.

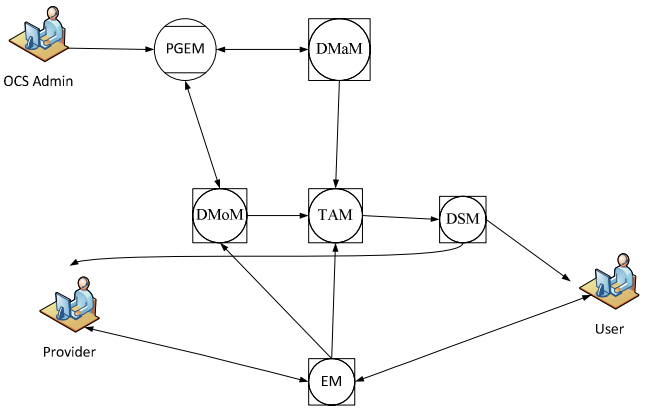

Fig. 2: OCS trust management systems architectural components

V. OCS TRUST MODEL VERIFICATION

Two usage scenarios are employed for demonstrating the applicability of the OCS trust model and the OCS trust management system. One of the usage scenarios will be on an IaaS and the other on SaaS. The verification process seeks to demonstrate the applicability of the trust model and trust management system to standard cloud computing services.

A. IaaS usage scenario

We look at the case where an OCS member has spare storage space at its data center and has virtualized this storage space to provide a storage service on the OCS platform. The standardized pSLA that may apply to such a service is show in table 1 below.

Table 1: pSLA template for IaaS

Attribute	value types
Service Identification	Service ID & category ID
Service Type / category	IaaS & category ID
Availability	50 % uptime
Service support	No
Service support type	N/A
Maintenance notification	Yes
SLA dependencies	{}
Service location	{}
Security	None
Data encryption	None
Privacy	None
Certification	{}

1) sSLA created from a pSLA

In this case, the service provider may for example, change only the availability level to 95%, and provide security support of data backup and recovery as shown in table 2.

Table 2: sSLA created from a pSLA

Attributes	value types
Service Identification	Service ID & category ID
Service Type / category	IaaS & category ID
Availability	95 % uptime
Service support	No
Service support type	N/A
Maintenance notification	Yes
SLA dependencies	{}
Service location	{}
Security	Data backup & recovery
Data encryption	None
Privacy	None
Certification	{}

2) rSLA created from sSLA or pSLA

A service user may accept this sSLA in order to use the service or request for additional requirements in its rSLA.

3) Computation of trust values

i. Extraction of critical rating vectors from the rSLA

If the user in interested in service availability, security and maintenance notification as its criteria for using then service, then the critical rating vector is given by

$$R_c^s(t_i) = \{availability, \sec urity, maintenance\ notification\}$$

ii. Extraction of users' utility function

Assuming the user's utility function increases monotonically with availability above 90% given that security and maintenance notification are provided, then the utility function is given by

$$U_c(R) \geq 90\%$$

iii. Approximation of reputational information into standardize probability distribution functions

We look at two cases, one in which the trustworthiness $\tau_c^s(R,t_i)$ approximates a uniform distribution function, and the second approximates a normal distribution. The empirically collected data is approximated to standard distribution functions using appropriate curve fitting algorithms as mentioned in Section IV.E.2 above.

We compute the trust level of the service when $\tau_c^s(R,t_i)$ is a uniform distribution with parameters $U(a,b)$

$$T_c^s(t_i) = \int_{0.9}^{1} \tau_c^s(R,t_i).dR = 1 - \int_{-\infty}^{0.9} \tau_c^s(R,t_i).dR = 1 - \Phi(\frac{0.9-a}{b-a})$$

a. We compute the trust level of the service when $\tau_c^s(R,t_i)$ is a normal distribution with parameters

$$N(\mu,\sigma) = \frac{1}{\sqrt{2*pi*\sigma^2}} * \exp(\frac{-(x-\mu)^2}{2*\sigma^2})$$

$$T_c^s(t_i) = \int_{0.9}^{1} \tau_c^s(R,t_i).dR = 1 - \int_{-\infty}^{0.9} \tau_c^s(R,t_i).dR = 1 - \Phi(\frac{0.9-u}{\sigma})$$

Fig. 3 shows the characteristic curves of the trust level against the lower limit in the above scenario when the upper limit of the availability is 100%. It shows for the uniform distribution and a normal distribution curve with standard deviations equal to that of the uniform distribution.

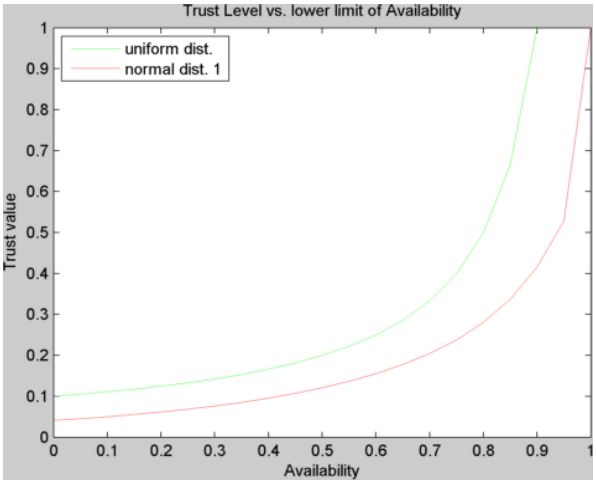

Fig. 3: Trust Level against varying Lower limit of availability

Similarly the trust level for the provider of the services is computed in the same way us above. Fig. 4 shows the composite trust level in our scenario with varying service weight, where the trust level of the service is from the uniform distribution as above and the trust level of the provider is the normal distribution with the same standard deviation as that of the uniform distribution.

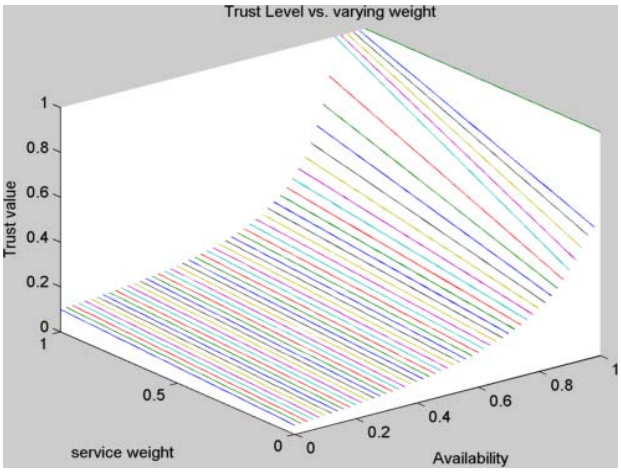

Fig. 4: Trust Level against varying service weight

Fig. 5 show the composite trust level in our scenario with varying user utility, where the trust level of the service is from the uniform distribution as above and the trust level of the provider is the normal distribution with the same standard deviation as that of the uniform distribution. Fig.5a shows when the service and the service provider have equal weight of 0.5 in the composite trust value, Fig.5b shows when the service has a weight of 1(provider has a weight of 0) in the composite trust value, and Fig.5c shows when the service has a weight of 0 (provider has weight of 1) in the composite trust value.

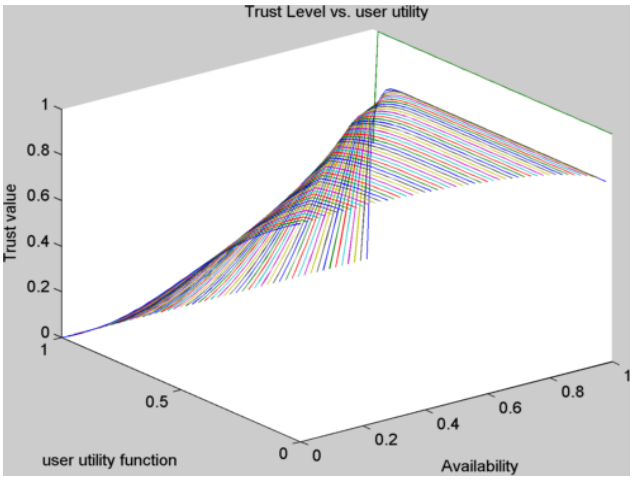

Fig. 5a: Trust level against varying user utility when the service and the service provider have equal weight of 0.5 in the composite trust value

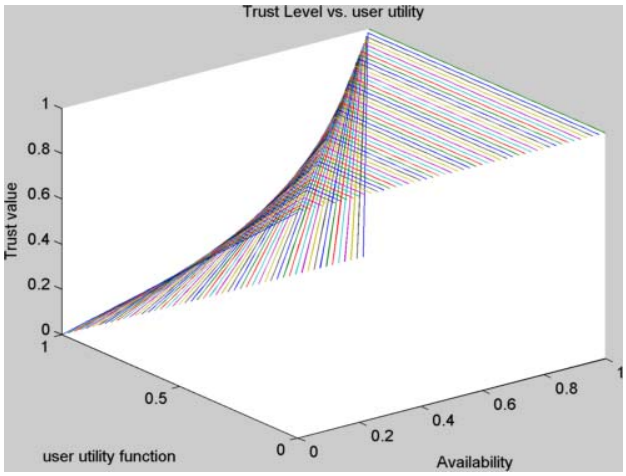

Fig. 5b: Trust level against varying user utility when the service has a weight of 1

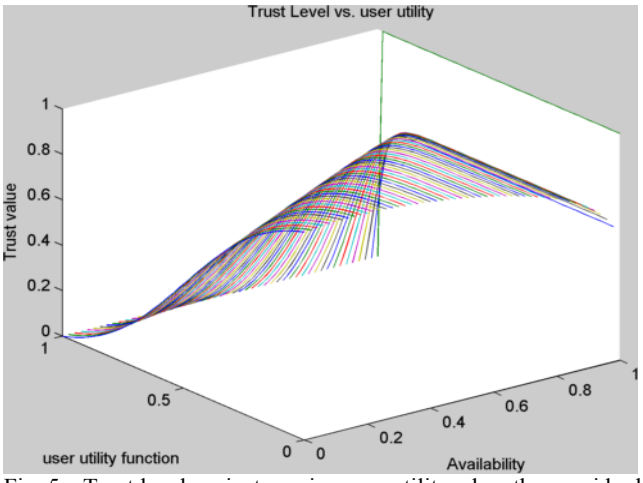

Fig. 5c: Trust level against varying user utility when the provider has a weight of 1

B. SaaS usage scenario

We now show the applicability of the trust management system in the case of a SaaS usage scenario. The only difference of the two processes is only in the pseudo SLA templates. Table 4 shows a pseudo SLA template for SaaS.

Table 3: pSLA for SaaS

Attribute	value types
Service Identification	Service ID &category ID
Service Type / category	SaaS & category ID
Availability	50 % uptime
Service support	No
Service support type	N/A
Maintenance notification	Yes
SLA dependencies	{}
Service location	{}
Security	None
Data encryption	None
Privacy	None
Certification	{}
Performance (Throughput)	1Kbps
Performance(Response time)	5sec

The user's utility increases monotonically with availability above 85% and response time below 2sec $R_c^s(t_i) = \{availability, \sec urity, response\_time\}$

1) sSLA created from pSLA

A service provider may accept this pSLA to create the sSLA for the service.

2) rSLA created from sSLA or pSLA

A service user may accept this sSLA in order to use the service or request for additional requirements in its rSLA.

The computation of the trust values follow the same process as is in the case of the IaaS above.

VI. CONCLUSION AND FUTURE WORK

This paper has looked at the design of a trust management system for OCS platforms. It has modeled trust for the OCS platform, designed a trust management system for OCS platforms, and verified the trust model and the trust management system through the simulation of the computation of the trust values with IaaS, and SaaS examples.

Even though our trust management systems contain the complete elements, we have focused mainly of the modeling of trust for the OCS platforms and the trust analysis components in our architecture. The other aspects require further work in terms of the implementation of the data monitoring and data management components. Secondly the decision support system and usability of the pseudo SLA templates in the system needs some further work for their verification. These further works will also require verifying the robustness and scalability of the trust management system.

REFERENCE

[1] E. Kuada and H. Olesen, "A Social Network Approach to Provisioning and Management of Cloud Computing Services for Enterprises," presented at the CLOUD COMPUTING 2011, The Second International Conference on Cloud Computing, GRIDs, and Virtualization, 2011, pp. 98–104.

[2] E. Kuada and H. Olesen, "Incentive mechanisms for Opportunistic Cloud Computing Services," in *2012 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2012, pp. 127 –136.

[3] E. Kuada, H. Olesen, and A. Henten, "Public Policy and Regulatory Implications for the Implementation of Opportunistic Cloud Computing Services for Enterprises," in *Workshop on Security in Information Systems*, Wroclav, 2012.

[4] E. Kuada, K. Adanu, and H. Olesen, "Cloud Computing and Information Technology Resource Cost Management for SMEs," in *Proceedings of IEEE Region 8 Conference EuroCon 2013*, University of Zagreb, Croatia, 2013, pp. 258 – 265.

[5] W. Jansen and T. Grance, "Guidelines on security and privacy in public cloud computing," *NIST Special Publication*, pp. 800–144, 2011.

[6] L. Badger, T. Grance, R. Patt-Corner, and J. Voas, "Draft cloud computing synopsis and recommendations," *NIST Special Publication*, vol. 800, p. 146, 2011.

[7] P. Mell and T. Grance, "The NIST definition of cloud computing (draft)," *NIST special publication*, vol. 800, p. 145, 2011.

[8] Z. Song, J. Molina, and C. Strong, "Trusted Anonymous Execution: A Model to Raise Trust in Cloud," in *2010 9th International Conference on Grid and Cooperative Computing (GCC)*, 2010, pp. 133 –138.

[9] M. Almorsy, J. Grundy, and A. S. Ibrahim, "TOSSMA: A Tenant-Oriented SaaS Security Management Architecture," in *2012 IEEE 5th International Conference on Cloud Computing (CLOUD)*, 2012, pp. 981 –988.

[10] W. Viriyasitavat and A. Martin, "Formal Trust Specification in Service Workflows," in *2010 IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC)*, 2010, pp. 703 –710.

[11] B. Grobauer, T. Walloschek, and E. Stocker, "Understanding Cloud Computing Vulnerabilities," *IEEE Security Privacy*, vol. 9, no. 2, pp. 50 –57, Apr. 2011.

[12] X. Zhang, H. Liu, B. Li, X. Wang, H. Chen, and S. Wu, "Application-Oriented Remote Verification Trust Model in Cloud Computing," in *2010 IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom)*, 2010, pp. 405 –408.

[13] J. Abawajy, "Establishing Trust in Hybrid Cloud Computing Environments," in *2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2011, pp. 118 –125.

[14] M. Kuehnhausen, V. S. Frost, and G. J. Minden, "Framework for assessing the trustworthiness of cloud resources," in *2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, 2012, pp. 142 – 145.

[15] C. Dellarocas, "The Design of Reliable Trust Management Systems for Electronic Trading Communities," in *SLOAN SCHOOL OF MANAGEMENT, MIT, 2000*, 2001.

[16] C. Shen, H. Zhang, H. Wang, J. Wang, B. Zhao, F. Yan, F. Yu, L. Zhang, and M. Xu, "Research on trusted computing and its development," *Sci. China Inf. Sci.*, vol. 53, no. 3, pp. 405–433, Mar. 2010.

[17] T. Parsons, *The Social System: (New York): The Free Press of Glencoe*. Collier-Macmillan, 1964.

[18] I. Breskovic, M. Maurer, V. C. Emeakaroha, I. Brandic, and S. Dustdar, "Cost-Efficient Utilization of Public SLA Templates in Autonomic Cloud Markets," in *Proceedings of the 2011 Fourth IEEE International Conference on Utility and Cloud Computing*, Washington, DC, USA, 2011, pp. 229–236.

[19] S. M. Habib, S. Ries, and M. Muhlhauser, "Cloud Computing Landscape and Research Challenges Regarding Trust and Reputation," in *2010 7th International Conference on Ubiquitous Intelligence Computing and 7th International Conference on Autonomic Trusted Computing (UIC/ATC)*, 2010, pp. 410 –415.

[20] G. Zhao, C. Rong, M. G. Jaatun, and F. E. Sandnes, "Reference deployment models for eliminating user concerns on cloud security," *J Supercomput*, vol. 61, no. 2, pp. 337–352, Aug. 2012.

[21] H. Salah and M. Eltoweissy, "Towards a personalized trust management system," in *2012 International Conference on Innovations in Information Technology (IIT)*, 2012, pp. 373 –378.